NAVAL POSTGRADUATE SCHOOL

MONTEREY, CALIFORNIA

THESIS

CONFIDENTIAL AND AUTHENTICATED COMMUNICATIONS IN A LARGE FIXED-WING UAV SWARM

by

Richard B. Thompson

December 2016

Thesis Advisor:	Preetha Thulasiraman
Second Reader:	John McEachen

Approved for public release. Distribution is unlimited.

REPORT DOCUMENTATION PAGE			*Form Approved OMB No. 0704–0188*
Public reporting burden for this collection of information is estimated to average 1 hour per response, including the time for reviewing instruction, searching existing data sources, gathering and maintaining the data needed, and completing and reviewing the collection of information. Send comments regarding this burden estimate or any other aspect of this collection of information, including suggestions for reducing this burden, to Washington headquarters Services, Directorate for Information Operations and Reports, 1215 Jefferson Davis Highway, Suite 1204, Arlington, VA 22202-4302, and to the Office of Management and Budget, Paperwork Reduction Project (0704-0188) Washington, DC 20503.			
1. AGENCY USE ONLY	**2. REPORT DATE** December 2016	**3. REPORT TYPE AND DATES COVERED** Master's thesis	
4. TITLE AND SUBTITLE CONFIDENTIAL AND AUTHENTICATED COMMUNICATIONS IN A LARGE FIXED-WING UAV SWARM			**5. FUNDING NUMBERS** R4KAW
6. AUTHOR(S) Richard B. Thompson			
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) Naval Postgraduate School Monterey, CA 93943-5000			**8. PERFORMING ORGANIZATION REPORT NUMBER**
9. SPONSORING /MONITORING AGENCY NAME(S) AND ADDRESS(ES) CRUSER			**10. SPONSORING / MONITORING AGENCY REPORT NUMBER**
11. SUPPLEMENTARY NOTES The views expressed in this thesis are those of the author and do not reflect the official policy or position of the Department of Defense or the U.S. Government. IRB Protocol number ____N/A____.			
12a. DISTRIBUTION / AVAILABILITY STATEMENT Approved for public release. Distribution is unlimited.			**12b. DISTRIBUTION CODE**

13. ABSTRACT (maximum 200 words)

Large unmanned aerial vehicle (UAV) swarms are a nascent technology promising useful military and civilian solutions to difficult problems. Securing data communications within the swarm is essential to accomplishing swarm objectives. The Naval Postgraduate School has successfully demonstrated the launch, flight and landing of 50 UAVs. The communications architecture to support a UAV swarm is unique. The practical challenges of creating a secure communications channel in the swarm are detailed in this thesis. The Advanced Encryption Standard (AES) was chosen as one of the encryption algorithms for testing, as it is authorized by the National Security Agency (NSA). Various modes of AES, including Galois/Counter Mode and Counter with Cipher Block Chaining Message Authentication Code, were analyzed within the swarm architecture. The impact of these authenticated encryption algorithms on network throughput and processor performance is presented. In addition to AES, ChaCha20-Poly1305, another type of authenticated encryption scheme, was studied. It was found to be the better solution for securing the swarm if classified data is not being handled or created.

14. SUBJECT TERMS unmanned aerial vehicle, UAV, security, swarm, communications, encryption, authentication			**15. NUMBER OF PAGES** 61
			16. PRICE CODE
17. SECURITY CLASSIFICATION OF REPORT Unclassified	**18. SECURITY CLASSIFICATION OF THIS PAGE** Unclassified	**19. SECURITY CLASSIFICATION OF ABSTRACT** Unclassified	**20. LIMITATION OF ABSTRACT** UU

NSN 7540-01-280-5500 Standard Form 298 (Rev. 2-89) Prescribed by ANSI Std. 239-18

THIS PAGE INTENTIONALLY LEFT BLANK

CONFIDENTIAL AND AUTHENTICATED COMMUNICATIONS IN A LARGE FIXED-WING UAV SWARM

Richard B. Thompson
Lieutenant Commander, United States Navy
B.S., Brigham Young University, 2007

Submitted in partial fulfillment of the
requirements for the degree of

MASTER OF SCIENCE IN ELECTRICAL ENGINEERING

from the

NAVAL POSTGRADUATE SCHOOL
December 2016

Approved by: Preetha Thulasiraman
Thesis Advisor

John McEachen
Second Reader

R. Clark Robertson
Chair, Department of Electrical and Computer Engineering

THIS PAGE INTENTIONALLY LEFT BLANK

ABSTRACT

Large unmanned aerial vehicle (UAV) swarms are a nascent technology promising useful military and civilian solutions to difficult problems. Securing data communications within the swarm is essential to accomplishing swarm objectives. The Naval Postgraduate School has successfully demonstrated the launch, flight and landing of 50 UAVs. The communications architecture to support a UAV swarm is unique. The practical challenges of creating a secure communications channel in the swarm are detailed in this thesis. The Advanced Encryption Standard (AES) was chosen as one of the encryption algorithms for testing, as it is authorized by the National Security Agency (NSA). Various modes of AES, including Galois/Counter Mode and Counter with Cipher Block Chaining Message Authentication Code, were analyzed within the swarm architecture. The impact of these authenticated encryption algorithms on network throughput and processor performance is presented. In addition to AES, ChaCha20-Poly1305, another type of authenticated encryption scheme, was studied. It was found to be the better solution for securing the swarm if classified data is not being handled or created.

THIS PAGE INTENTIONALLY LEFT BLANK

TABLE OF CONTENTS

LIST OF FIGURES

THIS PAGE INTENTIONALLY LEFT BLANK

LIST OF TABLES

THIS PAGE INTENTIONALLY LEFT BLANK

LIST OF ACRONYMS AND ABBREVIATIONS

AE	authenticated encryption
AES	Advanced Encryption Standard
CBC	Cipher Block Chaining
CBC-MAC	Cipher Block Chaining Message Authentication Code
CCM	Counter with CBC-MAC
GCM	Galois/Counter mode
GCS	Ground Control Station
GPS	Global Positioning System
IPv4	Internet Protocol version 4
IV	initialization vector
MAC	message authentication code
NASA	National Aeronautics and Space Administration
NIST	National Institute of Standards and Technology
NPS	Naval Postgraduate School
NSA	National Security Agency
NS3	Network Simulator 3
TCP	Transmission Control Protocol
UAV	unmanned aerial vehicle
UDP	User Datagram Protocol

ACKNOWLEDGMENTS

There are many who have selflessly dedicated time, talent and energy toward bringing this project to fruition. To my cohort in the Space Systems Engineering program, thank you for your comradery, patience and listening ears. To my professors, it is one my life’s great fortunes to have been taught by true masters.

I am truly indebted to the great academics, researchers and technicians at CRUSER. This thesis would not have been possible without their assistance. Specifically I would like to thank Dr. Duane Davis, Michael Day and CTT2 Edward Montoya.

To Dr. Preetha Thulasiraman, my advisor, I appreciate the guiding hand and expert oversight. Thank you for helping me to focus my efforts, pulling me back when needed and pushing me on the right path to produce a useful product.

Most especially, I would like to thank my radiant wife, Tammy, and my delightful children—Sadie, James and Penny—for their patience, support, love and advice. I pity those who do not know them.

THIS PAGE INTENTIONALLY LEFT BLANK

I. INTRODUCTION

A. NEED FOR UAV SECURITY

In 2011, the Iranians captured an unmanned aerial vehicle (UAV). The RQ-170 Sentinel was flying in Iranian territory and was not brought down with weapons. The loss of this single UAV proved to be an intelligence coup for the Iranians and an embarrassment to the United States [1]. Securing UAVs from loss, malfunction and exploitation is of utmost importance. Unfortunately, security is often an afterthought [2]. Systems are designed, proof of concepts are built, and only then is security given any consideration.

With the advent of ever-cheaper hardware and further development of unmanned systems technology, the ability to field large swarms of unmanned aerial vehicles has become a reality. On August 27, 2015, the Naval Postgraduate School set a world record by autonomously launching, flying and landing 50 UAVs concurrently [3]. This swarm of UAVs can autonomously and cooperatively work toward an objective. The UAV swarm can be controlled by a single operator, guiding it to perform a specific behavior. Specific behaviors include area search, point intercept, ordered transit and mass-ordered landing. The technology is nascent, and as such, the full range of its capabilities are yet to be discovered. While the future prospects are exciting, the nature of UAV swarms is distinct from single UAVs and presents unique security challenges that must be addressed before UAV swarms are released for use as an operational tool.

B. CURRENT RESEARCH ENVIRONMENT

The inability to solve the security problem has and will continue to have severe consequences for swarm deployment [1], [4]. In the last several years, researchers have studied the various cyber security threats that UAV swarms face. These threats have been identified, along with extensive risk assessments [5], [6].

While there is a consensus in the research community that UAV swarms are vulnerable to cybersecurity attacks, there has been little movement in identifying appropriate algorithms to facilitate a security architecture for UAV swarm communications. As large UAV swarms are just coming into existence, examples of security implementations in practice do not exist. This is especially true for large swarms. A swarm that is unable to operate securely is almost entirely useless in any military or civilian application.

C. THESIS CONTRIBUTIONS

Currently, there is no security architecture built into the UAV swarm at NPS. This stems from various factors, including cost and how UAV performance will be affected by expensive security computations.

In this thesis, we focus on the impact of communication security on the swarm. This includes both encryption and authentication. UAV to UAV communications is a distinguishing feature of swarms and, thus, is a unique attack vector. Authenticated encryption (AE) is designed to simultaneously protect both a message's privacy and authenticity. For classified information communications, we study the Advanced Encryption Standard (AES). AES has been approved and adopted by the National Security Agency (NSA) as the official cryptographic module for the transmission of secret and top secret information [7]. We implement four AES modes: Counter with Cipher Block Chaining Message Authentication Code (CCM), Galois/Counter Mode (GCM), Synthetic Initialization Vector (SIV) and EAX. In addition, we also implemented ChaCha20-Poly1305, an unstandardized AE algorithm. This is used as a baseline for securing unclassified swarm communications. We present results that show the impact of these algorithms on network throughput and execution time.

While the solution provided in this thesis is specific to the UAV swarm found at the Naval Postgraduate School, the results can be generalized to provide valuable insight to other types of swarms including terrestrial and satellite swarms. For example, the National Aeronautics and Space Administration (NASA) has experimented with sending groups of satellites into orbit that work together on a particular goal [8]. Protecting the communications of these satellites is essential to the accomplishment of its mission. Solutions that can be adapted to this and other communication security problems are provided in this thesis.

It must be noted that parts of this thesis have already been published by the author at the time of this writing [9].

D. THESIS ORGANIZATION

The remainder of the thesis is organized as follows. In Chapter II, the NPS swarm design and operation is presented. A security assessment is presented that details numerous methods of attacking a swarm. We also discuss communication security and how it is to be implemented in the swarm. Various AE techniques are outlined for consideration. A network traffic analysis, including the impact of AE on network throughput is presented in Chapter III. In Chapter IV, a comparison of AE techniques implemented on the ODroid processor is discussed. The percentage of time spent on cryptographic operations for each AE technique is calculated. In Chapter V, we conclude the thesis and propose future research directions.

THIS PAGE INTENTIONALLY LEFT BLANK

II. NPS SWARM SYSTEM ARCHITECTURE

A. CONCEPT OF OPERATIONS

The swarm architecture and its detailed operation can be found in [3] and are summarized here. UAVs launch at regular intervals of about 15 seconds and transit to a waiting area where they await a command from the swarm controller. The swarm controller has a set of predefined behaviors to choose from. After performing the set of defined behaviors, a command is sent to land. The swarm then sorts itself and lands in an orderly fashion.

Each UAV was built from low-cost commercially available components. A picture of the NPS UAV is shown in Figure 1.

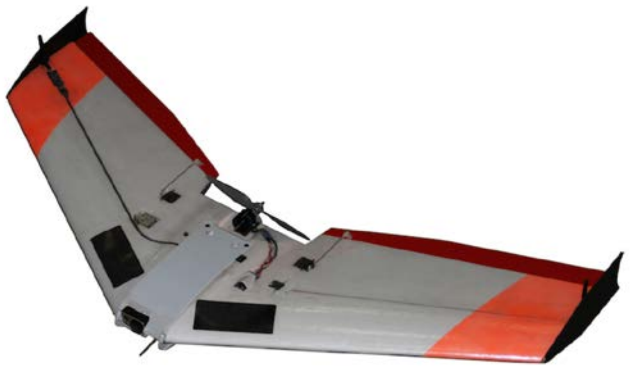

Figure 1. Picture of NPS UAV, Designed to be Low Cost Yet Capable. Source: [3].

The swarm communicates with an ALFA AWUS036NEH Long Range Wi-Fi Radio and processes information on an ODroid U3 computer with Ubuntu Linux 14.04. The ODroid computer has a Samsung Exynos4412 Prime Cortex-A9 Quad Core 1.7 GHz with 1 MB L2 cache. All software is coded in Python. Each UAV also possesses a remote control (RC) link and a 900-MHz serial link that are used for emergencies and not necessary for swarm operation. In a secure setting they are turned off.

B. COMMUNICATIONS ARCHITECTURE

The swarm communicates using IEEE 802.11n in ad hoc mode. The Wi-Fi radio has a single spatial stream operating on a 20-MHz channel, allowing for a maximum data rate of 72.2 Mbps. Each message is broadcast to each of the other UAVs. There is no expectation of privacy from any of the other UAVs. It is single hop, so there is no routing. All messages use the User Datagram Protocol (UDP) on top of the Internet Protocol version 4 (IPv4), which by definition is connectionless. There is a small subset of messages that do receive acknowledgements but is built on top of UDP with a custom protocol. The various information pathways to and from a UAV are shown in Figure 2.

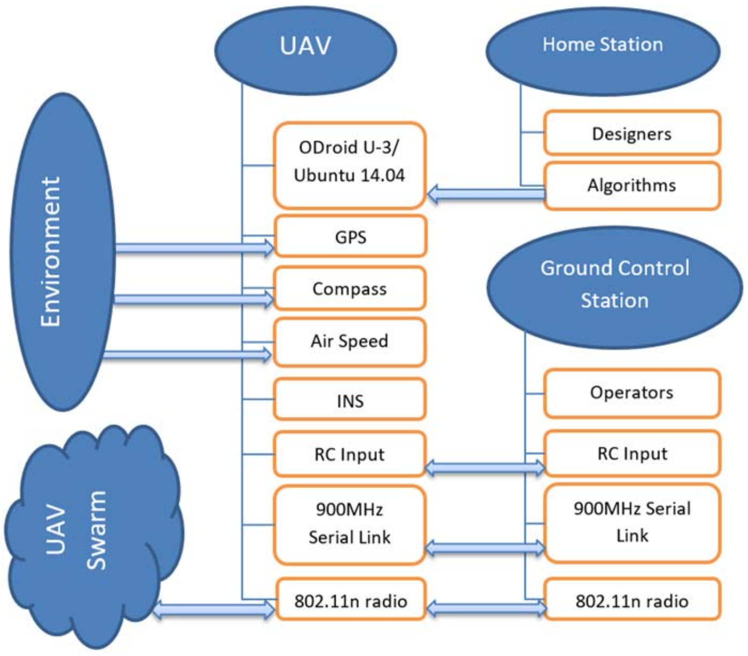

Figure 2. Illustration of UAV Hardware Configuration, Detailing Information Paths between Entities. Adapted from [1].

Increasing the reliability by using the Transmission Control Protocol (TCP), implementing a routing protocol, or communicating with direct links instead of broadcast will significantly increase latency and congestion on the network to the point that it becomes intolerable with a large swarm.

A side effect of the chosen architecture is the need to be tolerant of lossy communications. Messages must adhere to two principles. They must be stateless and idempotent. Stateless means a message cannot rely on a different message having been received. Idempotent means any message received may change the state of the UAV once and only once. Duplicate messages do not further change the UAV state [10].

The components that make up a message with typical lengths are listed in Table 1.

Table 1. Message Contents

Section	Typical Length (bytes)
Preamble	15
802.11 Header	34
IPv4 Header	20
UDP Header	8
Autonomous Capability Suite (ACS) Header	16
Encryption Overhead	Varies by Algorithm
Payload	Varies by Behavior

Any security implementation should not fundamentally alter or impose changes on the defined communication structure. Doing so would limit the range of behaviors available and increase complexity.

C. SWARM THREAT ASSESSMENT

Entry points into the UAV environment can be exploited by an adversary and must be guarded against. All the entry points to the UAV are shown in Figure 2. Each of the pathways and the common exploitations of that pathway are discussed in the following subsections. Possible mitigations are given for each exploitation. This section is not meant to be a complete list of possible attacks but to present the type of threat environment that exists and to demonstrate the vital need for AE.

1. Sensor Vulnerabilities

a. Magnetic and Air Speed Sensor

A picture of the GPS and magnetic sensor on the NPS UAV are shown in Figure 3. It is impractical to introduce environmental factors that significantly alter the input to the magnetic field or air speed sensors. Care must be taken to ensure the sensors are properly calibrated.

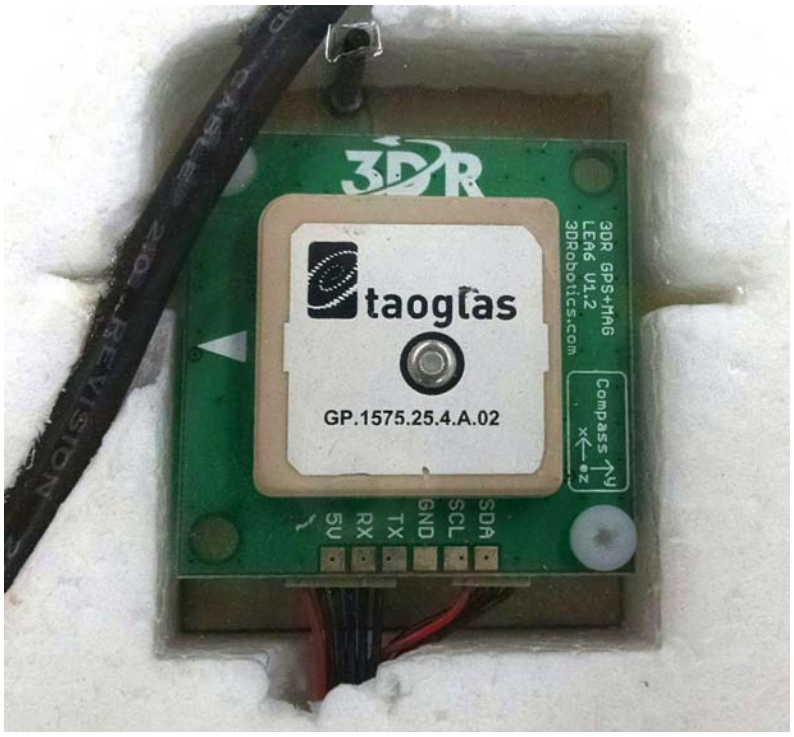

Figure 3. Picture of the GPS and Magnetic Sensor on the NPS UAV

b. GPS Sensor

The GPS sensor exploits and mitigations are detailed in Table 2.

Table 2. GPS Sensor Exploits and Mitigations

Exploit	Description	Possible Mitigations
GPS Spoofing	GPS satellite signal is overlaid by a stronger local signal from which it is used to direct the UAV.	- Dead reckoning with the assistance of magnetic sensors can be used to give a broad idea of where the UAV should go. If the UAV GPS receives a signal outside of the dead reckoning threshold, it should assume spoofing and return to base by reverse dead reckoning until it picks up on the base's authenticated homing signal. If available, terrain mapping could provide continuity of operations. Use of the P(Y)-code decreases the possibility of this exploit. - Monitor signal strength and identity [11]
GPS Jamming	GPS signal is overwhelmed by a stronger local signal	Same solutions as with GPS spoofing.

2. Communication Link Vulnerabilities

There are three communication pathways to the UAV. The first is a remote control input used for commanding the UAV one on one. The second is a 900MHz serial link between the ground station and autopilot. These first two are not used when conducting large swarm operations.

The third communication link is the IEEE 802.11n ad-hoc radio link. This is the communication link used when operating in a swarm environment and is the focus of this thesis. Various communication link exploits and mitigations that can be performed on each of the communication links are listed in Table 3.

Table 3. Communication Link Exploits and Mitigations

Exploit Name	Description	Possible Mitigations
Communication Capture/Eavesdropping	Information broadcasted to the Ground Control Station (GCS) such as telemetry or live video feeds are intercepted by the enemy.	Strong encryption, directional antennas.
Replay Attack	Packets are recorded and then rebroadcast at a later time. Attacker does not need to break encryption to perform this attack.	Strong authentication and serialization.
Man-in-the-Middle Attack	Information destined for a particular station is intercepted, modified and passed on to the destination. The link can be hijacked, eavesdropped on, or spoofed.	Strong encryption and authentication.
Denial-of-Service	A broad category of attack that congests the network with information such that the availability of the network is compromised.	- Directional antennas, multiple link frequencies. - Turn off applications like ICMP that can be used for such attacks.
Port Attacks	Accessing information through services such as TELNET and FTP. Viruses can be introduced or unintended programs can be run.	Close all unnecessary ports. Do not use insecure services.
RC Hijack	UAV can be forced to disconnect with controller, bad actor pairs with UAV.	Shut down RC link when not needed. Strong authentication between controller and UAV.
Side-Channel Attacks	Attacks that look at the pattern of RF radiation, timing of cryptographic operations or flight pattern to discover vulnerabilities.	- Perform random computer functions to throw off timing - Set timing of operations to all end at the same time.
Intelligence Gathering Vulnerabilities	Much information can be gathered just from the timing of the various messages and the different message sizes. Knowing what messages are being sent can be deduced through traffic analysis.	- Randomly vary message timing (especially for the repeating messages). - Appending a variable length tail, or ensuring all messages are the same length. - Periodically skip heartbeats, especially during periods when other messages are being sent.

3. GCS and Home Station Vulnerabilities

Various components on the NPS UAV are shown in Figure 4. Of particular note is the ODroid computer on the right side of the picture.

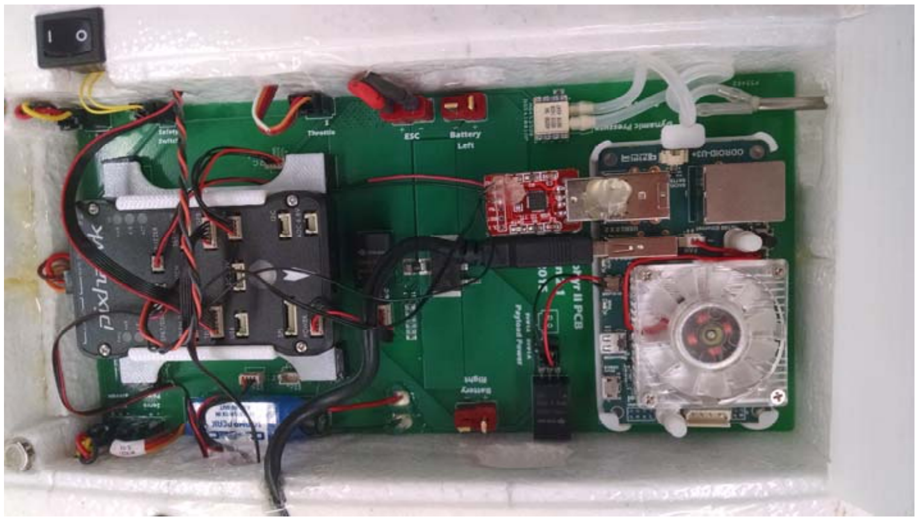
Figure 4. NPS UAV Motherboard Including ODroid

GCS and home station exploits and mitigations are listed in Table 4.

Table 4. GCS and Home Station Exploits and Mitigations

Exploit	**Description**	**Possible Mitigations**
GCS Virus Attack	A virus in a Ground Control Station could cause loss of sensitive information or loss of UAV control.	GCS networks should be closed, with no Internet connectivity. All software added to network should be thoroughly scrubbed.
Human Attackers	Malicious or incompetent operators sabotage the system by introducing vulnerabilities.	- Two person control of sensitive data and algorithms. - Access logs. - Information Assurance training.
Algorithm Exploits	Malicious actors can subtly manipulate algorithms to insidiously direct the swarm, or share false data to cause confusion, collisions and mayhem.	- Strong authentication. - Avoiding overt behaviors that betray a particular algorithm inasmuch as it is possible.

4. Hardware Vulnerabilities

A list of various hardware exploits and mitigations is given in Table 5.

Table 5. Hardware Exploits and Mitigations

Exploit	**Description**	**Possible Mitigations**
Technical malfunction	Malfunction of input sensors cause UAV to inadvertently land, possibly in enemy territory.	- Data destruct mechanism. - Return to base on sensor malfunction. - Land or Hover [1]. - Thorough testing.
Malicious Hardware Component	Hardware component might be tampered with [12].	Adopt an "Accept All" approach, where mechanical and peripherals are presumed malicious until proven otherwise [13].
Downed UAV Recovery	A crashed UAV might be recovered by an adversary. Data and algorithms on hard drives could be compromised.	- Critical algorithms and data at rest should be encrypted. - Store critical algorithms and data in non-persistent memory. - Create a data destroy routine when a crash has been detected. -General anti-reverse engineering solutions.

5. Summarization of Security Exploits and Mitigations

Naturally, proper risk assessment should be taken into consideration before applying any mitigation. Many of the mitigations might not make sense for a UAV swarm that is used for academic purposes on a secure military base.

It is absolutely clear that AE is necessary to truly protect a swarm. Due to the fact that swarm communications are broadcast in every direction, any receiver within range of the UAV is able to collect communications. In order to prevent tactical information from being discovered, encryption is a necessary requirement. Due to the ease of message capture and open nature of communications, the swarm is particularly vulnerable to replay attacks. As such, strong message authentication and serialization is also an absolute necessity. There is a cost, but that cost is unavoidable in almost any operational setting and must be accounted for.

It should also be noted that AE is not sufficient to completely protect the communication channel from all types of attacks. Attacks that target the availability of the communications network will not be deterred by AE. While an important area of study, countering availability attacks is beyond the scope of this thesis.

D. COMMUNICATIONS SECURITY ARCHITECTURE

There are two broad categories for providing encryption and authentication in the swarm: symmetrically and asymmetrically. With asymmetric encryption, a public key is used to encrypt and a private key is used to decrypt. This creates a different communication channel between each UAV. In addition, to provide authentication, the system requires a secure central repository for public keys. That repository is either a UAV or a ground station.

Asymmetric cryptography is not an option for swarm communications. It is a potential option for initially keying or for inflight rekeying, but for regular communications it permanently imposes an undesirable structure on the swarm. Symmetric encryption is faster and works within the current swarm architecture. In this construct the same key is preloaded before flight into each UAV. The UAV encrypts and decrypts with this same key.

1. Authenticated Encryption Alternatives

The only two options authorized for encryption by the National Institute of Standards and Technology (NIST) are the Advanced Encryption Standard (AES) and Triple Data Encryption Standard (3DES) [14]. It is well established that AES is faster, more secure and efficient [15] and is the only NSA approved symmetric encryption algorithm that can be used for encrypting top secret data [7]. Within AES, there are two AE modes available, GCM and Counter with CCM. GCM has the benefit of being both efficient and parallelizable [16]. While both CCM and GCM provide secure solutions, GCM has a reputation for being faster [17].

SIV and EAX are highly specialized AE techniques designed for specific problems. Initialization vectors (IVs) are used by cryptographic algorithms to ensure duplicate messages produce unique ciphertexts. For most algorithms, including CCM and

GCM, improper selection of the IV has catastrophic results. SIV was designed to be tolerant of IV misuse but sacrifices speed [18].

EAX is another AES mode that was built to improve certain features of CCM. It can use arbitrary length IVs and does not need to know the message length in advance before beginning the algorithm. It also is a two-pass mode (i.e., one pass to achieve privacy and the other pass for authenticity) and, as such, was not designed for speed [19]. SIV and EAX are currently under consideration by NIST and were included in the ODroid processor performance analysis [20].

The ChaCha20 algorithm for encryption and Poly1305 for authentication have become a popular alternative in industry for performing AE [21]. In 2014, Google replaced GCM on its Android phones with ChaCha20-Poly1305, believing it to be more secure and showing it to be significantly faster in software implementations [22]. ChaCha20-Poly1305 was designed to be fast in software on generic computer architectures by minimizing hardware intensive operations such as matrix multiplication [23]. While not approved for classified data, it was included in the analysis to provide both a baseline and an option for secure communications when the swarm is not performing classified operations.

The AES algorithms were implemented using the Python library PyCryptodomex 3.4.2. This library is written in Python except for the pieces critical to performance, which were written in C [24]. ChaCha20-Poly1305 was implemented using the Python library PyNaCl 1.0.1. The library is also written in Python but is a wrapper around the libsodium library, which is written in C.

PyCryptodomex allows for IV sizes of between 7 and 13 bytes for CCM. For CCM, the chosen IV size is 13 bytes and for all other AES based algorithms is 16 bytes as recommended by [25]. The IV size for PyNaCl is required to be 24 bytes [26].

Message authentication codes (MACs) are bytes attached to each message used to verify the authenticity of a message. The MAC size for each algorithm was 16 bytes [25], [26].

2. Appropriate Layer for Applying Security

In most applications there is a choice of whether to place security at the application layer or at the transport layer. Due to the connectionless nature of the swarm, it is possible to place security at an even lower level, just above the Physical Layer. Every message that is sent on the swarm channel is received by every other entity, with no expectation of routing; therefore, the entire frame can be encrypted, including MAC addresses. This prevents an adversary from gathering potentially critical message source information. Doing so, however, requires specialized hardware and/or software and is not possible with the current swarm configuration.

Security at the Transport Layer is also problematic. WPA2 with authentication on Ubuntu 14.04 is not supported for a connectionless ad-hoc network without a central access point or centralized authentication server [27]. The ALFA Wi-Fi radio provides WPA2 encryption but only with 128-bit keys and lacks authentication that will work with the connectionless swarm configuration [28].

Security at the Application Layer is easily implementable and provides privacy and authenticity. For the purposes of this study, security was implemented at the Application Layer on the ODroid processor. Implementation at the Application Layer allows us to determine the impact of security on communications within the swarm.

E. CHAPTER SUMMARY

The UAV swarm architecture imposes constraints on the security framework. There are numerous ways to attack the swarm. To provide a secure channel for swarm communications without modifying the swarm architecture requires symmetric encryption with a shared key between all UAVs. In the remainder of this thesis, the cost associated with implementing the various AE alternatives mentioned in this chapter are detailed.

THIS PAGE INTENTIONALLY LEFT BLANK

III. EFFECTS OF AUTHENTICATED ENCRYPTION ON THE NETWORK THROUGHPUT

A. SIMULATION DESCRIPTION

To determine the effects of security on the network, it is necessary to get a sense of the amount and type of traffic that is being passed.

Three instances of UAVs were created using multi-UAV simulation-in-the-loop (SITL) software, commanded and flown with actual flight software as described in [29]. The messages passed between UAVs and ground station were the same as though the UAVs had been flying. For each message on the channel, the message type, size, time of transmission and sending UAV was recorded. A larger software swarm would have been preferred, but hardware limitations resulted in inaccurate results when the number of UAVs grew larger than three. In Section C of this chapter, how the results obtained from a three UAV swarm are relevant to larger swarms are discussed.

The swarm has the following behaviors that it can perform:

- **Line Formation:** UAVs form into a line and fly to a designated location or flight pattern.
- **Swarm Search:** UAVs cooperatively search a specified area.
- **Greedy Shooter:** UAVs find the closest enemy UAV and tag it as being shot.
- **PN Interceptor:** Command given to one UAV to intercept another.
- **Eager Altitude Sort:** UAVs are sorted by altitude. Missing information is requested from other UAVs. Responses to requests are given about itself and other missing UAVs.
- **Lazy Altitude Sort:** Similar to Eager Altitude Sort except only missing information about itself is broadcast.
- **Independent Transit:** All UAVs in a subswarm transit separately to a geographic position.
- **Sequence Land:** UAVs land in an orderly fashion.

A software application called *Swarm Commander* is used to command the swarm to perform these behaviors during actual operations. In the simulation, *Swarm Commander* was used to execute each of the behaviors in the order presented earlier. When parameters were required, the default values were selected.

B. SIMULATION RESULTS

There was an average of 13.02 messages per UAV transmitted in any given second. The average number of messages per second for a three UAV swarm was 39.07, with a standard deviation of 6.618 and a high of 52. The average unencrypted message size was 141.61 bytes with a standard deviation of 12.54. The throughput of the 3-UAV swarm over the time of the experiment is illustrated in Figure 5. As can be seen, throughput is fairly constant, and it is difficult to distinguish when certain behaviors are occurring.

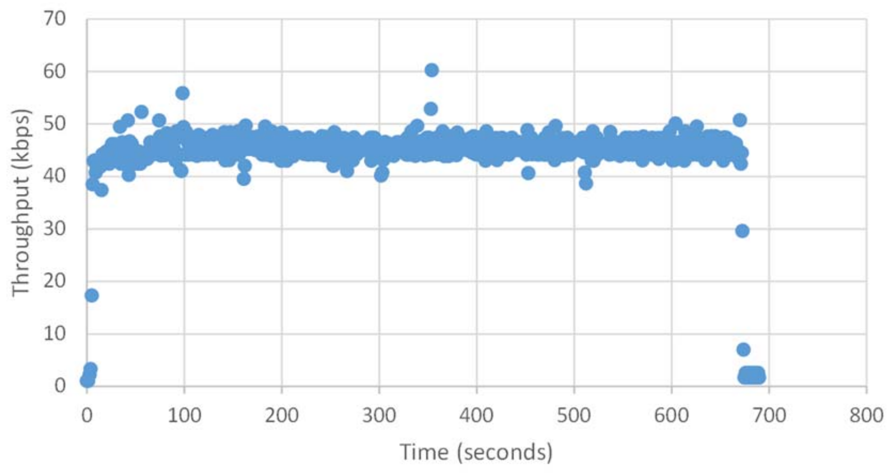

Figure 5. Total Throughput on a 3-UAV Swarm Channel without AE

Traffic is dominated by just three types of messages: Pose, Flight Status and Heartbeat. These messages are used by the UAVs to update each other and the ground station with telemetry and health information. They are sent out at regular intervals regardless of swarm size. With a 3-UAV swarm, the breakdown of occurrence by message type is shown in Table 6.

Table 6. Occurrence of Message Type

Message Type	Occurrence
Pose	74.38%
Flight Status	14.38%
Heartbeat	9.48%
Other	1.76%

The throughput results of a 3-UAV swarm when Pose, Flight Status and Heartbeat messages are removed is shown in Figure 6. Throughput is much lower, and areas where behaviors occur are somewhat distinguishable when compared to Figure 5.

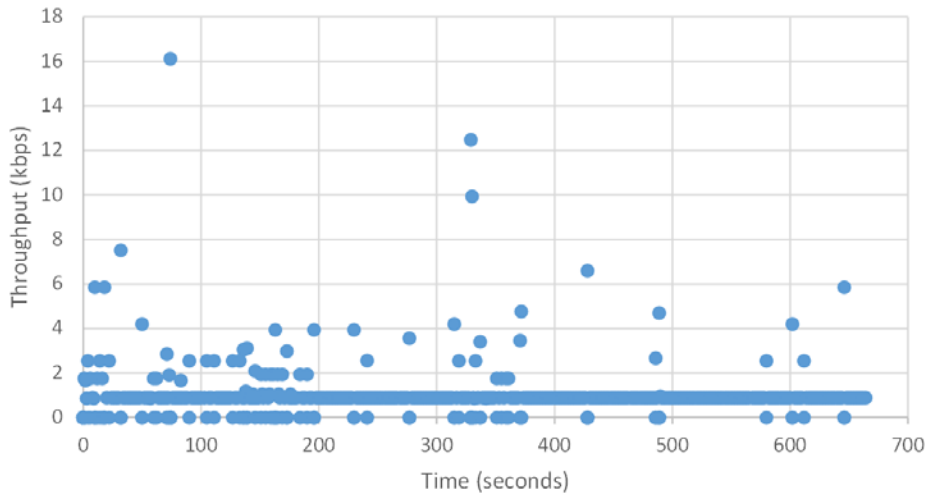

Figure 6. Total Throughput on a 3-UAV Swarm Channel with Pose, Flight Status and Heartbeat Messages Removed

The maximum and average throughputs that were achieved with each AE method tested are listed in Table 7. In addition, the average bytes per message overhead incurred due to cryptographic operations is also listed. ChaCha20-Poly1305 has the greatest overhead. Note that the bytes per message overhead incurred by each algorithm is constant, regardless of message length; thus, as message length increases, the overhead as a percentage of message length due to cryptographic operations decreases.

Table 7. Effects of Cryptographic Operations on Throughput

Cryptographic Algorithm	Average Throughput (kbps)	Maximum Throughput (kbps)	Average Overhead Incurred (%)
None	44.23	60.28	0
CCM	53.29	71.64	20.4
GCM	54.29	72.82	22.5
ChaCha20-Poly1305	56.73	76.96	28.2

C. EXTENDING RESULTS TO LARGER SWARMS

Knowing how these results apply to larger swarm sizes is essential for an accurate understanding of the effects of AE on network throughput. To determine the effects, it was necessary to analyze each message in order to determine how it depends on the size of the swarm. There are two ways in which a message can be dependent on swarm size: length dependent and frequency dependent. A message is length dependent if a particular message changes length as a function of swarm size. A message is frequency dependent if how often a message is sent depends on swarm size.

In a best case scenario, with complete message independence of swarm size, the growth rate of network traffic is $O(n)$, where *n* is the number of UAVs in the swarm. In a behavior where messages grow by a constant amount with each additional UAV, the growth rate of network traffic is $O(n^2)$. In a behavior where the frequency of messages increases at a constant rate with each additional UAV, the growth rate of network traffic is also $O(n^2)$. In a worst case scenario, where both message frequency and length grow by a constant amount with each additional UAV, the growth rate of the network traffic is $O(n^3)$.

When all messages are independent of swarm size, how average throughput increases as the swarm size grows, extrapolated based on the data gathered in the 3-UAV swarm simulation, is illustrated in Figure 7. In this scenario, even in a 100-UAV swarm with 30% cryptographic overhead, the swarm does not approach the 72.2-Mbps throughput ceiling.

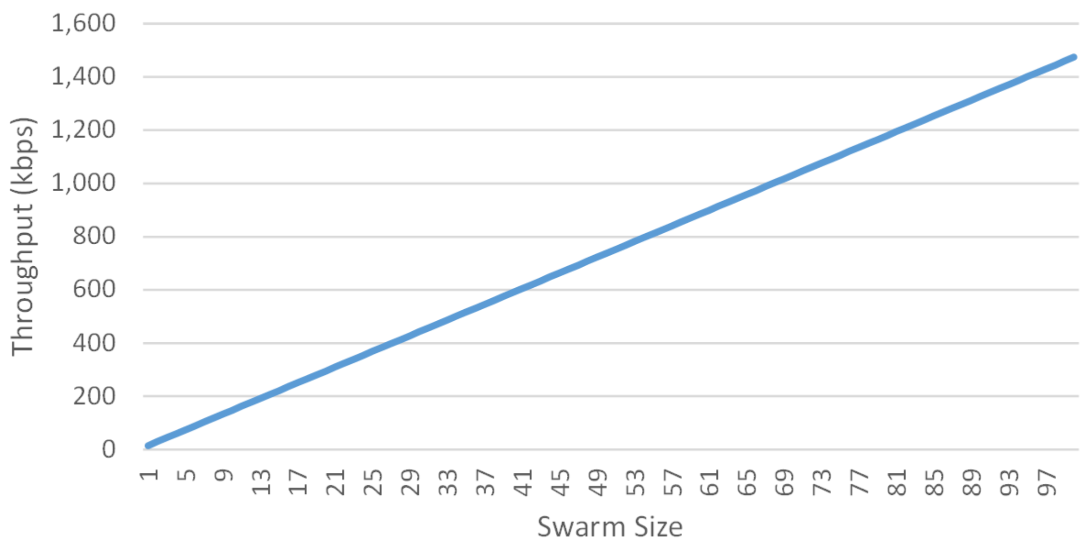

Figure 7. Average Total Throughput as a Function of Swarm Size Where Messages Do Not Depend on Swarm Size (No AE)

In contrast, if an additional packet is sent by each additional UAV per second and each of those messages grows by 10 bytes for each UAV, we get the results shown in Figure 8. With larger swarms the channel is overwhelmed, even without cryptographic overhead.

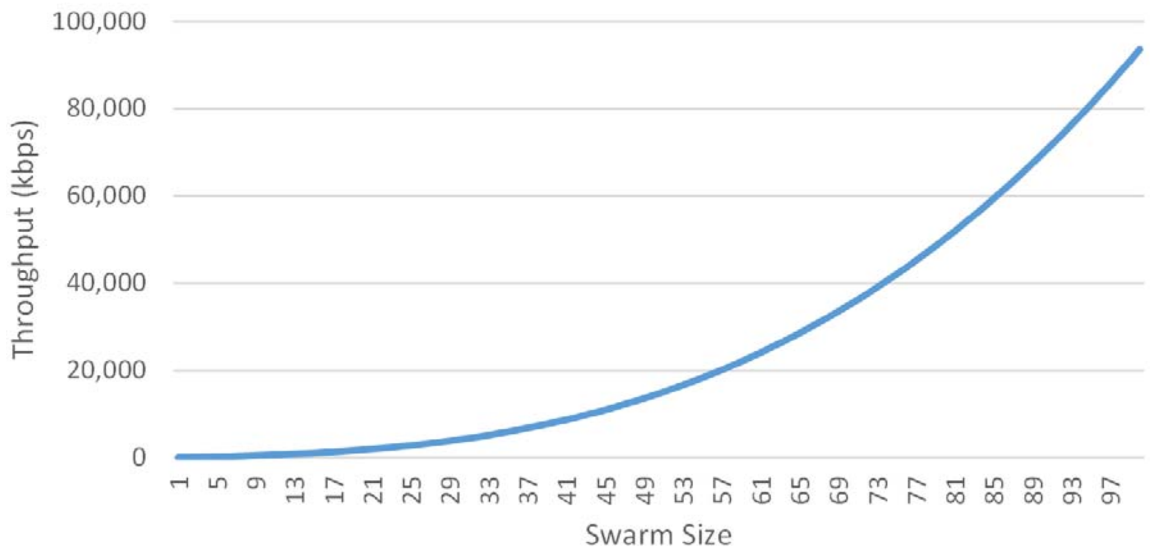

Figure 8. Average Total Throughput as a Function of Swarm Size Where Messages Have Both Length and Frequency Dependence on Swarm Size (No AE)

In Figure 7 and Figure 8, a reasonable lower and upper bound, respectively, on network traffic is shown. To determine the likelihood of the two scenarios, a closer examination of each behavior was undertaken. How each behavior affects message dependence on swarm size is shown in Table 8.

Table 8. Message Dependence on Swarm Size

Behavior/ Message Type	**Length Dependence**	**Frequency Dependence**
Flight Status	No	No
Pose	No	No
Heartbeat	No	No
Eager Altitude Sort	Yes	Yes
Greedy Shooter	No	Yes
Independent Transit	No	No
Lazy Altitude Sort	No	Yes
Line Formation	Yes	Yes
PN Interceptor	No	No
Sequence Landing	No	No
Swarm Search	No	Yes

Flight Status, Pose and Heartbeat messages are sent out at frequencies of 10.0 Hz, 2.0 Hz and 2.0 Hz, respectively, regardless of swarm size.

The Eager Altitude Sort and Line Formation behaviors make use of a consensus sort algorithm. This algorithm requests a message from each UAV from which it lacks information. A larger swarm increases the likelihood of missing information. In addition, response messages from any UAV includes information from any other UAV whose information was also requested and message lengths also increase. Designers recognized this and limited the frequency of the messages to 4.0 Hz, thus, the frequency dependence has an upper bound.

In the Greedy Shooter behavior, the frequency dependence is very weak and probably undetectable. As the swarms grow, the density of UAVs also grows, creating more shooting opportunities and, thus, more kill reports.

In the Lazy Altitude Sort behavior, the consensus sort algorithm is also used with the difference being any UAV whose information is requested only responds with information about itself. Thus, there is no length dependence. The frequency dependence is again limited to 4.0 Hz.

In the Swarm Search behavior, a lead UAV is designated at the commencement of the behavior. The lead UAV proceeds to assign search areas to each of the other UAVs in its subswarm. There is frequency dependence but only from the point of view of the lead UAV.

The worst case for network traffic is during a behavior using the consensus sort algorithm where each UAV requests information from every other UAV. If an additional four messages are sent from each UAV per second, each UAV grows ten bytes for each UAV in the swarm, and a 30% overhead is added on for worst case cryptography. The resulting throughput is similar to that shown in Figure 9.

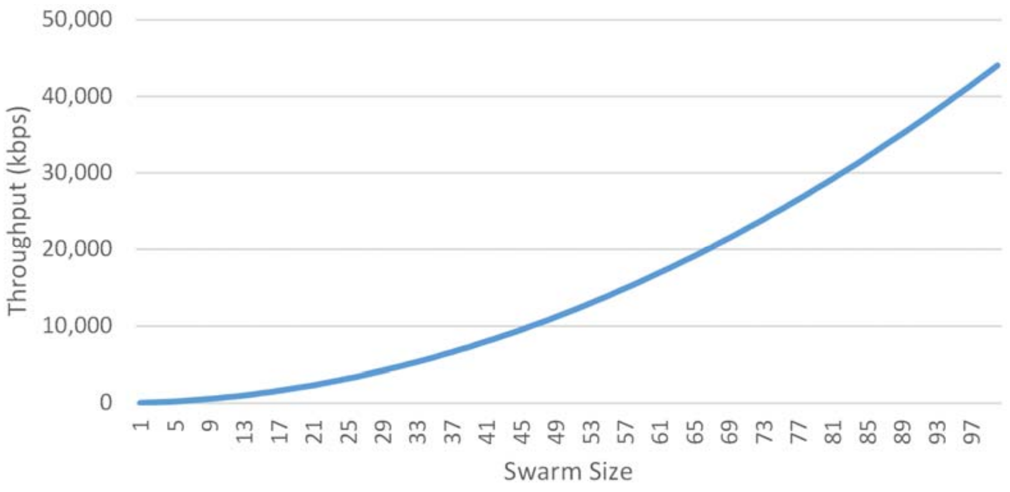

Figure 9. Expected Throughput of a Worst Case Scenario Behavior, with Worst Case Cryptographic Overhead Included

Even with the worst case behavior, and using the largest cryptographic overhead, the swarm communications channel is left with some operational margin.

D. CHAPTER SUMMARY

The effects of AE on network throughput were discussed in this chapter. The amount of traffic on the communication channel was found to be very dependent on the swarm behavior. There are a number of possible behaviors that the swarm can perform. Even for worst case behavior while performing the AE algorithm with the largest overhead, the swarm throughput appears to have margin to spare on a 72.2-Mbps channel. A discussion of how AE impacts processing performance is contained in the next chapter.

IV. EFFECTS OF AUTHENTICATED ENCRYPTION ON THE ODROID COMPUTER

A. EXPERIMENT DESCRIPTION

To completely understand the impact of AE on the swarm, it is necessary to determine if the ODroid processor is capable of sustaining the cryptographic overhead incurred. To make this determination, GCM, CCM, SIV, EAX and ChaCha20-Poly1305 algorithms were implemented and executed on the ODroid hardware.

AE was performed on messages of sizes ranging from eight to 32,768 bytes and then decrypted and authenticated. For each message size, this step was repeated 10,000 times and averaged. Each message was randomized on each pass to create as cold a cache as possible. The experiment was repeated with key sizes of 128, 192 and 256 bits. ChaCha20-Poly1305 and SIV do not provide functionality for key sizes of 128 and 192 bits.

B. EXPERIMENT RESULTS

The execution time for each AE algorithm with key sizes of 128, 192, and 256 bits, are shown in Figures 10, 11 and 12, respectively.

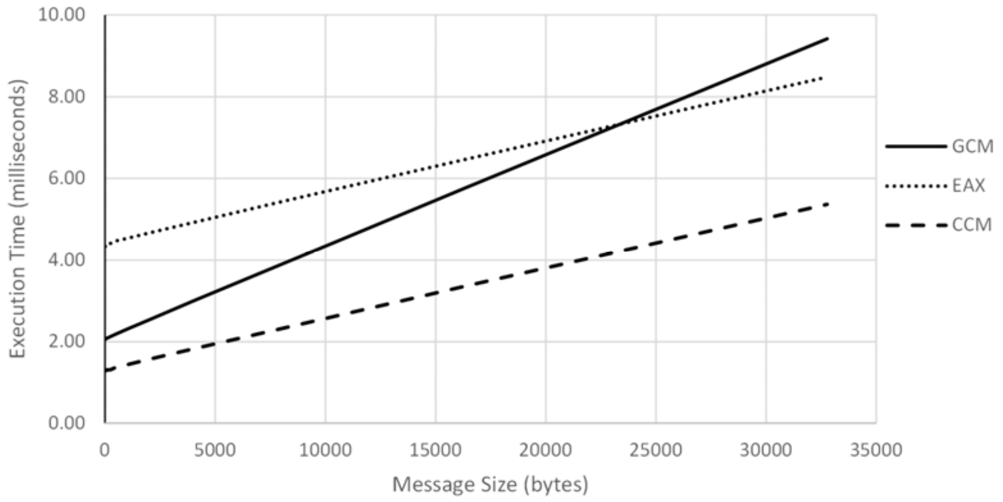

Figure 10. ODroid Performance While Executing Cryptographic Operations with 128-Bit Key Sizes

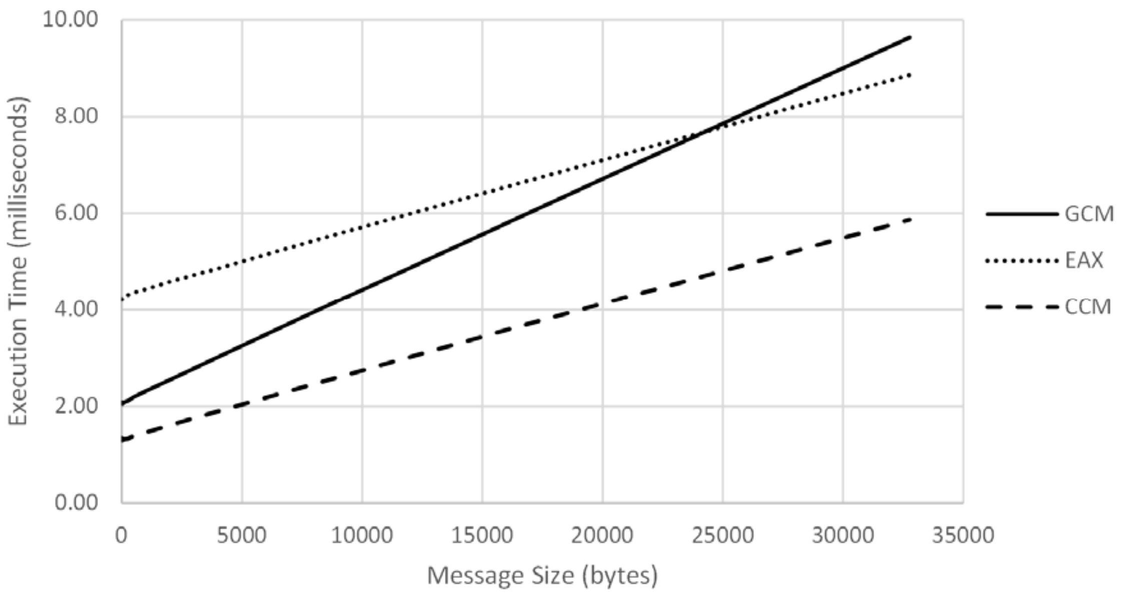

Figure 11. ODroid Performance While Executing Cryptographic Operations with 192-Bit Key Sizes

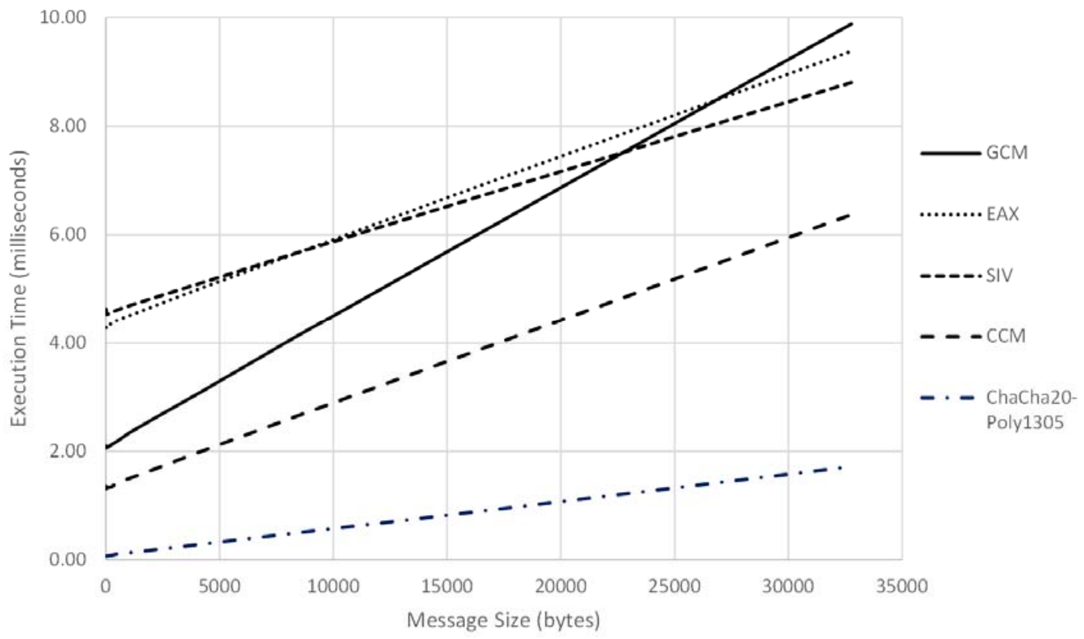

Figure 12. ODroid Performance While Executing Cryptographic Operations with 256-Bit Key Sizes

As was expected, the best performance for any key size was ChaCha20-Poly1305. EAX and SIV had the poorest overall results. This poor performance was anticipated given their specialized nature.

Based upon the results, CCM outperformed GCM. As stated earlier, the biggest advantage of GCM is its ability to be parallelizable. GCM did not perform better than CCM in this situation for the following reason: GCM performs best when software is tailored to the hardware, making use of parallel processors [16], [30]. It does not appear that the PyCryptodomex library makes full use of the parallelizable nature of GCM and certainly was not designed specifically for the ODroid computer. These results are consistent with the results found in [19].

Another interesting result is how little key size affects speed of execution, especially with smaller messages. The average length of an unencrypted message was 141.61 bytes. The average time (in ms) to execute each iteration of an average length message is listed in Table 9.

Table 9. Cryptographic Times for Average Sized Message in ms

	128 Bit Key	**192 Bit Key**	**256 Bit Key**
CCM	1.30	1.31	1.32
ChaCha20-Poly1305			.0781
EAX	4.40	4.28	4.33
GCM	2.10	2.08	2.09
SIV			4.54

C. ODROID PERFORMANCE UNDER VARIOUS NETWORK LOADS

From the channel throughput analysis in Section IV, we can predict the burden that cryptography places on the ODroid computer. Given that the average unencrypted message size was 141.61 bytes and the average number of messages per second per UAV was 13.02, an average cryptographic load for a given swarm size on the ODroid computer can be estimated.

Assuming messages do not exhibit any dependence on swarm size (as laid out in Figure 7) and assuming an average case throughput of 13.02 messages per UAV per second, we see the percentage of time out of each second spent conducting cryptographic operations is displayed in Figure 13.

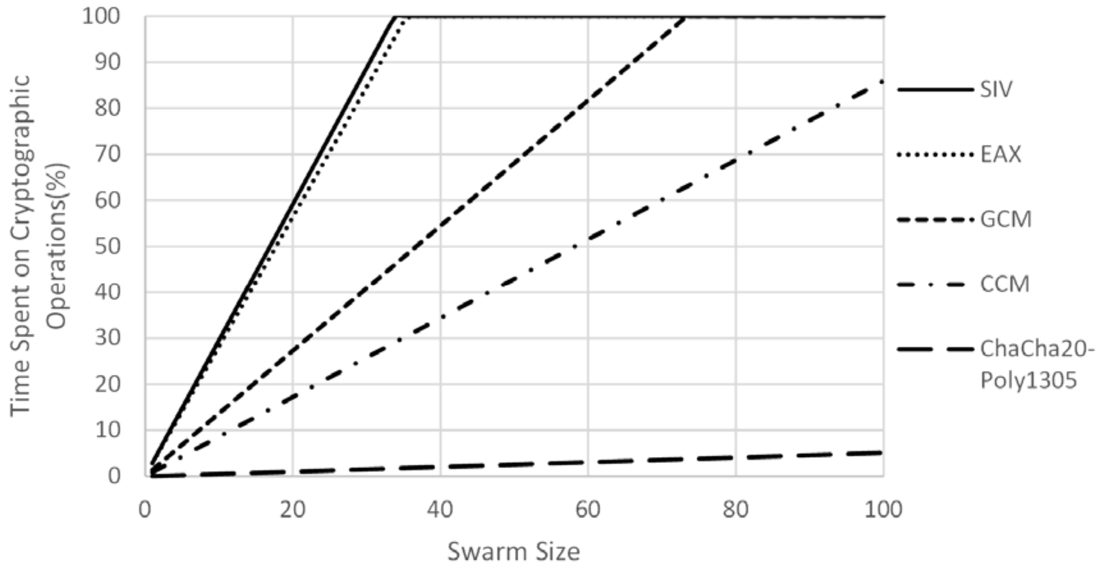

Figure 13. Average Case Percentage of Time Spent on Cryptographic Operations with 256-Bit Key

SIV and EAX are not able to support a 50-UAV swarm, as the ODroid would be spending 100% of its time on cryptographic operations and still not be able to keep up with the traffic load. GCM and CCM might be able to manage an average case load in a 50-UAV swarm but would be consuming a significant amount of the processing capacity, leaving little to other processes. The worst case scenario as defined by Figure 9 is shown in Figure 14. Using a smaller key size does not improve performance by any meaningful amount.

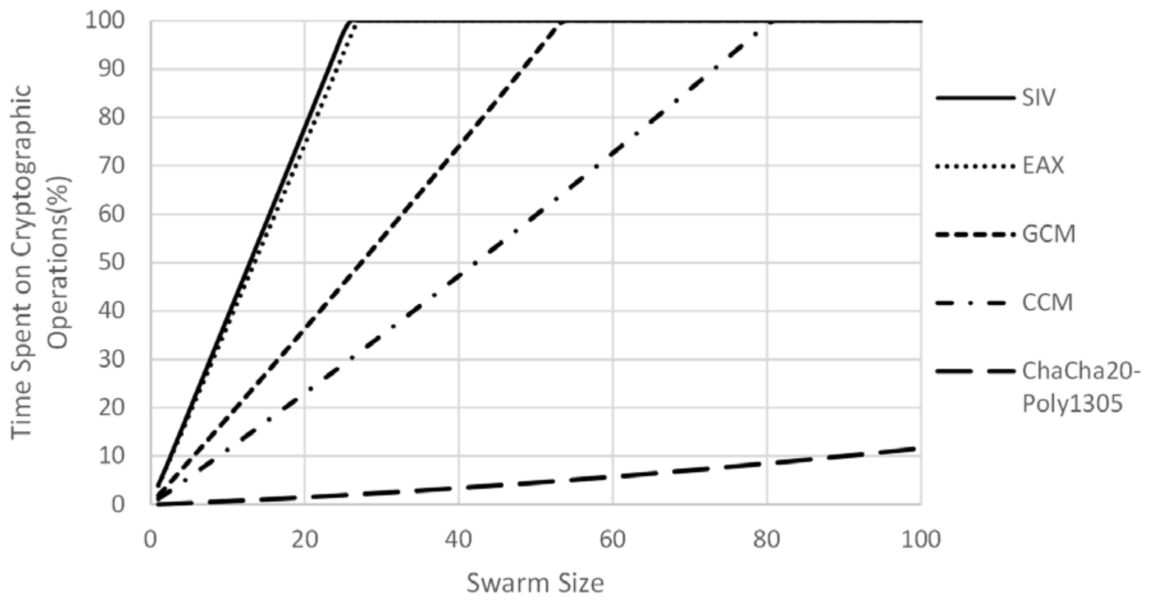

Figure 14. Worst Case Percentage of Time Spent on Cryptographic Operations with 256-Bit Key.

Clearly SIV, EAX, GCM and CCM are not an option. It does appear that ChCha20-Poly1305 would be successful in providing AE for the swarm.

D. MITIGATIONS FOR CLASSIFIED INFORMATION

Currently, the swarm at NPS is used for academic purposes. It does not gather or create classified information and does not require the use of an AES-based algorithm. In the event that classified information was gathered, the following mitigations can be used to enable the use of either GCM or CCM:

- Upgrade to a more powerful processor.
- Use an Application Specific Integrated Circuit (ASIC).
- Tailor the AES algorithm to the ODroid processor.
- Only use AE on command data.

Performing AE only on command data gives access to information about the state of the swarm and its current location and might not be tolerable. It also opens the risk of an adversary planting false telemetry data and surreptitiously changing the state of the swarm. It would, however, prevent the taking over of the swarm by the sending of direct commands. The impact on swarm performance in the worst case network channel scenario presented in Figure 9 when only command data is encrypted is shown in Figure 15.

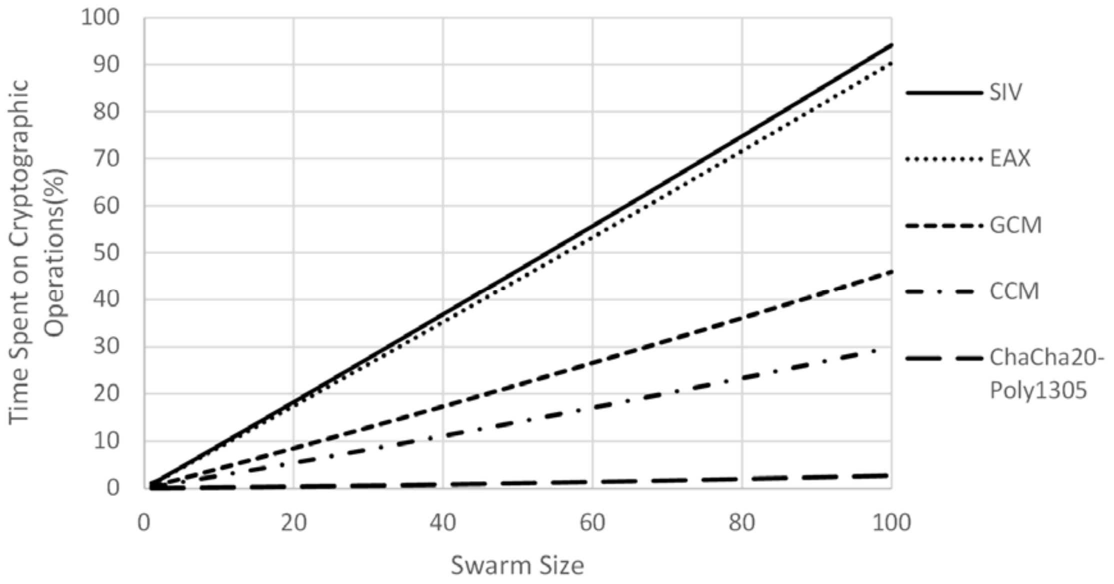

Figure 15. Worst Case Percentage of Time Spent on Cryptographic Operations Only on Command Data with 256-Bit Key.

With this mitigation, SIV and EAX still fall short of acceptability on large swarms. GCM and CCM perform tolerably, and ChaCha20-Poly1305 again proves the superior method.

V. CONCLUSION

A. SUMMARY

As UAV swarms come into practice, security should be treated as any other subsystem and included as part of the initial design. Doing so upfront allows other subsystems to be designed to handle the cost incurred by security. In any swarm, AE has an impact on performance. In this thesis, that impact on the very specific NPS UAV swarm was investigated.

In the current swarm configuration and architecture, analyses indicates that performing AE with either GCM, CCM, SIV or EAX is not feasible. GCM and CCM are only feasible by accepting risk. The best choice by far is ChaCha20-Poly1305 and should be the AE choice in any scenario where classified data is not being handled or created.

Operating a swarm without communication security poses too great a risk in almost any operation. Completion of this work is essential for the ultimate success of UAV swarms in any operational environment.

B. FUTURE WORK

In this thesis, baseline encryption and authentication algorithms to deal with communication link vulnerabilities were established and are a stepping stone to developing a comprehensive security architecture for the UAV swarm system. This was the first step, and there is much more work to be done in this area of research. Future work is needed to fully and accurately determine the true affect of AE on swarm communications. The following subsections detail courses of action to further measure the impact of AE on a swarm.

1. Network Simulator 3 (NS3) Model

NS3 is software used to model a network. The NPS UAV swarm can be modeled in NS3. This will allow a more in depth understanding of how much capacity is actually available, and we will be able to model how distance, packet collision avoidance and modulation affect throughput.

2. Traffic Bombardment

An experiment designed to bombard an ODroid with typical traffic through an IEEE 802.11n connection should be executed. The amount of traffic should be increased until a saturation point is discovered. This will determine how much processor margin is available for cryptographic operations.

3. Power Consumption

Power consumption due to AE could prove significant. Determining the impact of encryption on battery life is essential. To determine this, UAVs performing AE and others not performing AE should have their battery life determined and compared against each other.

4. Implementation in the NPS Swarm

To truly and fully understand the impact of AE on performance, it must be actually implemented in the swarm. This should be the ultimate goal of future researchers.

APPENDIX A. PYTHON CODE FOR PERFORMING AUTHENTICATED ENCRYPTION

A. CODE DESCRIPTION

The following code performs various cryptographic operations. This module is created for academic purposes in order to determine effect of encryption and authentication on the swarm. It has not been reviewed by a cryptographic expert. Known vulnerabilities are

- NONCE reuse is improbable, but possible.
- There is no check for replay attacks. Part of the nonce could include a sequence number. Messages should be checked against this sequence number.

The functions encrypt_AES and decrypt_AES use AES to provide application layer encryption and authentication. The functions encrypt_fast and decrypt_fast use the NaCl library to provide rapid encryption. AES is very secure but might be an overdesign. NaCl uses ChaCha20-Poly1305 for AE operations.

Included is code that tests the correct operation of various functions and also code that determines the execution time of various cryptographic operations. To use this module properly, it should be called from the acs_socket.py module. All messages make use of the acs_socket.py module to communicate external to the UAV, so it is an ideal location to place cryptographic operations. In particular, there is a send function and a receive function where cryptographic operations should take place; thus, to secure all communications leaving the UAV through the IEEE 802.11n interface, only two functions in the acs_socket.py module should be modified.

B. SOURCE CODE OF ACS_CRYPTOGRAPHY.PY

```
#!/usr/bin/env python
#-------------------------------------------------------------------
# ACS Encryption Library
# LT Richard Thompson, 2016
#-------------------------------------------------------------------
from Cryptodome.Cipher import AES
from Cryptodome.Random import get_random_bytes
from uuid import getnode as get_mac_addr
import nacl.secret
import nacl.utils
import struct

encryption_on = True
key = b’Sixteen byte keySixteen byte key’ #two sixteen byte keys make a 256 bit key


if encryption_on:
	print (“Encryption On”)
else:
	print (“Encryption Off”)

AESmode = AES.MODE_GCM
box = nacl.secret.SecretBox(key)

# For documentation on parameter sizes for different modes
# refer to API: http://legrandin.github.io/pycryptodome/Doc/3.3.1/Crypto.Cipher-
module.html
AESmacSize = 16
if AESmode == AES.MODE_CCM:
	AESnonceSize = 13
else:
	AESnonceSize = 16

def encrypt(data, msg):
	#return _encrypt_AES (data)
	return _encrypt_fast(data)

def decrypt(data):
	#return _decrypt_AES (data)
	return _decrypt_fast(data)

def _encrypt_AES (msg):
	nonce = get_random_bytes(AESnonceSize)
	cipher = AES.new(key, AESmode, nonce)
	encrypted_msg = cipher.encrypt(msg)
	mac = cipher.digest()
	fmt = “>“+str(AESnonceSize)+”s”+str(AESmacSize)+”s” +
str(len(encrypted_msg)) + “s”
	return struct.pack(fmt, *(nonce, mac, encrypted_msg))
def _decrypt_AES (msg):
	nonce = msg[0 : AESnonceSize]
	mac = msg[AESnonceSize : AESmacSize+AESnonceSize]
	ciphertext = msg[AESmacSize+AESnonceSize : ]
	cipher = AES.new(key, AESmode, nonce)

	try:
		plaintext = cipher.decrypt_and_verify(ciphertext, mac)
		return plaintext
	except ValueError:
		print (“Key incorrect or message corrupted. acs_cryptography.py decrypt
function\n”)
		return (“ERROR in acs_cryptography.py”)


def _encrypt_fast(msg):
	nonce = nacl.utils.random(nacl.secret.SecretBox.NONCE_SIZE)
	encrypted = box.encrypt(msg, nonce)
	return encrypted


def _decrypt_fast(msg):
	try:
		# box.decrypt also authenticates
		plaintext = box.decrypt(msg)
		return plaintext
	except:
	 print (“Key incorrect or message corrupted. acs_cryptography.py decrypt_fast
function\n”)
	return (“ERROR in acs_cryptography.py”)

#### Test Cases
## General Test
msg = b’This tests the encrypt/decrypt functions’
print (‘\n\nBefore: %s’ % msg)
mycipher = encrypt(msg)
print (mycipher)
print (‘After: %s’ % decrypt(mycipher))

## Test encrypt AES
msg = b’Testing encrypt_AES/decrypt_AES: AES.MODE_GCM. This is the message.’
print (‘\n\nBefore: %s’ % msg)
mycipher = _encrypt_AES(msg)
print (mycipher)
print (_decrypt_AES(mycipher))

## Test encrypt_fast
```

```
msg = b’Testing encrypt_fast/decrypt_fast: This is the message.’
print (‘\n\nBefore: %s’ % msg)
mycipher = _encrypt_fast(msg)
print (‘During: %s’ % mycipher)
print (‘After: %s’ % _decrypt_fast(mycipher))

#### Timing Test
import time
import sys
import os
import math

iters = 10000
AESmodes = [AES.MODE_CCM, AES.MODE_GCM, AES.MODE_EAX,
AES.MODE_SIV]
AESmodesTostr = [“CCM,” “GCM,” “EAX,” “SIV”]
for i in range(0, 13):
        print (“Message Size in bytes: “+ str(int(math.pow(2,i))*8))
        # AES_Modes
        for j in range(0, len(AESmodes)):
                cryptoTime = 0
                AESmode = AESmodes[j]
                if AESmode == AES.MODE_CCM:
                        AESnonceSize = 13
                else:
                        AESnonceSize = 16
                for x in range(0, iters):
                        msg = os.urandom(int(math.pow(2,i))*8)
                        start = time.clock()
                        mycipher = _encrypt_AES(msg)
                        _decrypt_AES(mycipher)
                        end = time.clock()
                        cryptoTime = cryptoTime + (end - start)
                print (str(AESmodesTostr[j]) + “: “ + str(cryptoTime/iters))

        # NaCl
        cryptoTime = 0
        for x in range(0, iters):
                msg = os.urandom(int(math.pow(2,i))*8)
                start = time.clock()
                mycipher = _encrypt_fast(msg)
                _decrypt_fast(mycipher)
                end = time.clock()
                cryptoTime = cryptoTime + (end - start)
        print (“NaCl: “ + str(cryptoTime/iters))
```

APPENDIX B. SUGGESTED CONFIGURATION OF EQUIPMENT USED TO PERFORM TRAFFIC BOMBARDMENT EXPERIMENT

A. DESCRIPTION OF EXPERIMENT

To test how a UAV handles an increasing amount of traffic, the swarm can be configured as shown in Figure 16. In this configuration, a Swarm Traffic Simulator can generate an increasing amount of traffic. The actual UAV running the actual payload is connected through an Ethernet cable to a different laptop computer feeding it simulated flight information. Software on the UAV tracks how many packets it manages to receive, decrypt and authenticate successfully. Repeating the experiment without AE and comparing throughput will clarify the impact of AE on the swarm.

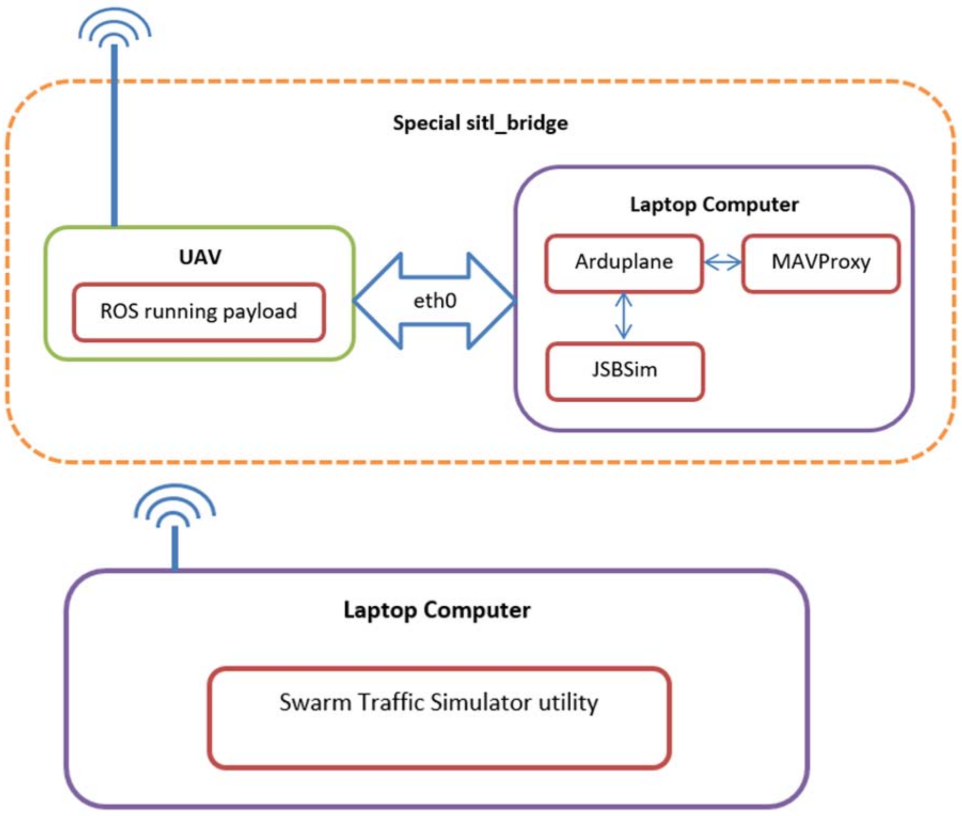

Figure 16. Approximated Hardware-in-the-Loop Simulation.

B. EQUIPMENT CONFIGURATION

1. **Turn on the o-droid**
   a. The blue light should be flashing on the wifi adapter. If a constant light, reboot.
2. **The o-droid must be configure to run DHCP**
   ssh into the o-droid on the UAV
   sudo vim /etc/network/interfaces
   change iface eth0 static to be dhcp (not on wlan0)
3. **Connect the laptop and o-droid through ethernet**
   a. Run an ethernet cable from the odroid to a switch connected to a DHCP server
      i. If no DHCP server, you will have to mess with static IPs
   b. Run an ethernet cable from your laptop to the same switch
4. **Setup wifi**
   a. Connect a wifi network adapter to both the o-droid and the laptop
   b. Run ifconfig and see what interface they are on.
5. **Configure laptop to be on zephyr adhoc wifi network**
   a. wifi_config.sh -2 wlan3 223 (-2 is the red team network, -3 is blue team, check to see what network it is on. wlan3 is the interface that the highpower wifi adapter is using)
      i. Check ifconfig that it has an ip adress on wlan3 (or the relevant interface)
      ii. Ping the UAV at 192.168.3.XX (Where XX is the UAV number, a sticker on the plane, .2 is the blue team, .3 is the read team [yes, it is backwards])
6. **ssh into the odroid**
   a. ssh odroid@192.168.3.XX
   b. password is odroid
7. **On laptop, run special multisitl with Arduplane, mavproxy and JSBSim**
   a. modify (if you haven't already) ~/ACS/acs-env/multi-sitl-start.bash
      i. add the word exit
   b. ~/multi-sitl-start-no-payload.bash 1
      i. its configured to have an "exit" before adding payload
      ii. We only want one instance of it
8. **On the o-droid, start the ROS startup script**
   a. roslaunch ap_master sitl.launch id:=101 name:=sitl101 sitl:=tcp:172.20.90.181:6772 port:=5554 ns:=sitl101 dev:=eth0
   b. For the tcp address, use the ETHERNET address (run ifconfig on the laptop). Don't change the port after the tcp address.
9. **To Run the Health monitor**
   a. Run it over wifi, not ethernet, because the ethernet connection is being used.

LIST OF REFERENCES

[1] K. Hartmann and C. Steup, “The vulnerability of UAVs to cyber attacks—an approach to risk assessments,” in *Proc. of 5th International Conference on Cyber Conflict*, 2013, pp. 78–100.

[2] N. Ferguson, B. Schneier and T. Kohno, *Cryptography Engineering* 1st ed. Indianapolis, IN: Wiley, 2010.

[3] T. H. Chung, M. R. Clement, M. A. Day, K. D. Jones, D. Davis and M. Jones, “Live-fly, large-scale field experimentation for large numbers of fixed-wing UAVs,” in *IEEE International Conference on Robotics and Automation (ICRA)*, Stockholm, Sweden, 2016.

[4] M.S. Faughnan, B. J. Hourican, G. C. MacDonald, M. Srivastava, J. A. Wright, Y. Y. Haimes, E. Andrijcic, Z. Guo and J. C. White, “Risk analysis of unmanned aerial vehicle hijacking and methods of its detection,” in *Proc. of IEEE Systems and Information Engineering Design Symposium*, 2013, pp. 145–150.

[5] K. Mansfield, T. Eveleigh, T. H. Holzer and S. Sarkani, “Unmanned aerial vehicle smart device ground control station cyber security threat model,” in *Proc. of IEEE International Conference on Technologies for Homeland Security*, 2013, pp. 722–728.

[6] A. Javaid, W. Sun, and M. Alam, “UAVSim: a simulation testbed for unmanned aerial vehicle network cyber security analysis,” in *Proc. of IEEE Globecom Workshop-Wireless Networking and Control for Unmanned Autonomous Vehicles*, 2013, pp.1432-1436.

[7] Commercial national security algorithm suite. (2015, Aug. 19). National Security Agency. [Online]. Available: https://www.iad.gov/iad/programs/iad-initiatives/cnsa-suite.cfm

[8] J. Hanson, J. Chartres, H. Sanchez and K. Oyadomari, “The EDSN intersatellite communications architecture.” in *28th Annual AIAA/USU Conference on Small Satellites*. Logan, Utah, 2014.

[9] R. Thompson and P. Thulasiraman, “Confidential and authenticated communications in a large fixed-wing UAV swarm,” in *Proc. of IEEE Network Computing and Applications*, Boston, MA, 2016, pp. 375–382.

[10] M. Clement, private communication, Dec. 2015.

[11] S. Giray, “Anatomy of unmanned aerial vehicle hijacking with signal spoofing,” in *Proc. of 6th International Conference on Recent Advances in Space Technologies*, Istanbul, Turkey, 2013, pp. 795–800.

[12] Z. Birnbaum, A. Dolgikh, V. Skormin, E. O’Brien and D. Muller “Unmanned aerial vehicle security using recursive parameter estimation,” in *Proc. of International Conference on Unmanned Aircraft Systems*, Orlando, FL, 2014, pp. 692–702.

[13] D. F. Pigatto, G. F. Roberto, L. Goncalves, J. F. R. Filho, A. S. R. Pinto and K. R. L. J. C. Branco, “HAMSTER-healthy, mobility and security-based data communication architecture for unmanned aircraft systems,” in *Proc. of International Conference on Unmanned Aircraft Systems*, Orlando, FL, 2014, pp. 52–63.

[14] A. Vassilev, “Annex A: approved security functions for FIPS PUB 140–2, security requirements for cryptographic modules,” National Institute of Standards and Technology, Gaithersburg, MD, 2016.

[15] H. O. Alanazi, B. B. Zaidan, A. A. Zaidan, H. A. Jalab, M. Shabbir and Y. Al-Nabhani, “New comparative study between DES, 3DES and AES within nine factors,” *Journal of Computing*, vol. 2, no. 3, pp. 152–157, 2010.

[16] M. Dworkin, “NIST Special Publication 800–38D: Recommendation for Block Cipher Modes of Operation: Galois/Counter Mode (GCM) and GMAC,” National Institute of Standards and Technology, Gaithersburg, MD, 2007.

[17] D. A. McGrew and J. Viega, “The security and performance of Galois/Counter Mode (GCM) of operation,” in *Progress in Cryptology—INDOCRYPT*, Chennai, India, 2004.

[18] P. Rogaway and T. A. Shrimpton, “A provable-security treatment of the key-wrap problem.” in *Advances in Cryptology*, St. Petersburg, Russia. 2006, vol. 4004 of LNCS, pp.373-390.

[19] Švenda, P. (2016) “Basic comparison of modes for authenticated-encryption (IAPM, XCBC, OCB, CCM, EAX, CWC, GCM, PCFB, CS).” [Online]. Available: https://www.fi.muni.cz/~xsvenda/docs/ AE_comparison_ipics04.pdf

[20] Modes Development. (2016, Mar. 24). National Institute of Standards and Technology. [Online]. Available: http://csrc.nist.gov/groups/ST/toolkit/BCM/ modes_development.html#01

[21] N. Sullivan. (2015, Feb. 23). Do the ChaCha: better mobile performance with cryptography [Online]. Available: https:// blog.cloudflare.com/do-the-chacha-better-mobile-performance-with-cryptography/

[22] E. Bursztein. (2014, Apr. 24). Speeding up and strengthening HTTPS connections for Chrome on Android [Online]. Available: https://security.googleblog.com/2014/04/speeding-up-and-strengthening-https.html

[23] D. Bernstein. “The Salsa20 family of stream ciphers” in *New Stream Cipher Designs*, 2008, pp. 84–97.

[24] pycryptodomex 3.4.1 (n.d.) Cryptographic library for Python. Python Package Index. [Online]. Available: https://pypi.python.org/pypi/ pycryptodomex/3.4.2. Accessed Aug. 8, 2016.

[25] Module AES. (2016 Feb. 7) Pycryptodome.org. [Online]. Available: http://legrandin.github.io/pycryptodome/Doc/3.4/Crypto.Cipher.AES-module.html#MODE_CCM

[26] D. J. Bernstein, T. Lange and P. Schwabe, “The security impact of a new cryptographic library.” in International Conference on Cryptology and Information Security in Latin America, Santiago, Chile, 2012, pp.159-176.

[27] WifiDocs/Adhoc. (2011, May 23). Ubuntu documentation. [Online]. Available: https://help.ubuntu.com/community/WifiDocs/Adhoc

[28] AWUS036NEH. (n.d.) ALFA Network. [Online]. Available: https://www.alfa.com.tw/products_show.php?pc=34&ps=22. Accessed Aug. 11, 2016.

[29] M. A. Day, M. R. Clement, J. D. Russo, D. Davis and T. H. Chung, “Multi-UAV software systems and simulation architecture,” in *International Conference on Unmanned Aircraft Systems (ICUAS)*, Denver, Colorado, 2015.

[30] K. Thompson, “Bad APIs Cause Bugs,” in *Zero Bugs and Program Faster*, 1st ed. Seattle: Kate Thompson Books, 2015, ch. 29, pp. 61–64.

THIS PAGE INTENTIONALLY LEFT BLANK

INITIAL DISTRIBUTION LIST